Hacer esta prueba

**Feature Engineering**

Para correr XGBoost se realizo el siguiente Feature Engineering

• **Lags**, valores previos de una determinada serie de tiempo. Se consideraron valores desde 1 a 36.

* **Delta lags**, diferencia entre los valores de la variable 'lag’. Se consideraron valores desde 1 a 35. Tener en cuenta que los extremos no agregan valor por contener todas sus filas nulas.
* **Medias móviles** Promedio de los últimos periodos consecutivos de la serie temporal. Se consideraron valores de 1 a 36.
* **Minimos** Booleana indicando si dicho valor era el mínimo para ese conjunto de datos en determinado periodo. Se consideraron valores de 1 a 36.
* **Maximos** Booleana indicando si dicho valor era el máximo para ese conjunto de datos en determinado periodo. Se consideraron valores de 1 a 36.
* **fechas** Transformaciones relacionadas a la predictora del periodo para extraer: año, mes, cantidad de días en el mes, quarter, continuidad temporal por seno y coseno.
* **variables exogenas** Cotización del dólar en Argentina, Índice de Precios al Consumidor (IPC), receso escolar, mes con feriados, anomalías políticas
* **Prophet** Se usaron las variables (tendencia, estacionalidad, máximos, rangos, términos aditivos, términos multiplicativos) creadas por el modelo de Prophet como input para el modelo de XGBoost. Es importante destacar que dado que dichas variables están expresadas en toneladas se tuvo que incluir un paso previo para aplicar la misma transformación de escalado correspondiente a cada serie agrupada por producto
* **Clusters** En una instancia posterior se aplicaron algoritmos de clustering (DTW) con el fin de encontrar la similitud entre series. Finalmente se incluyeron también como una nueva variable al dataset.

Show less

[10:26](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154009)

**Clase**

Ratio (tn / tn+2)  
Otra variante fue la de crear un ratio entre la cantidad de toneladas totales vendidas ese periodo y la cantidad en los dos periodos posteriores con el fin de capturar el porcentaje de cambio y también evitar los problemas de escala de las series. Fue importante la manipulación final del dataset para reconstruir las predicciones basados en el último dato del periodo 201912 y aplicar el ratio tanto para 202001 como 202002.

[10:26](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154010)

**Preprocesamiento**

Dentro de las tareas de preprocesamiento, y por la versatilidad del modelo elegido no fue estrictamente necesario realizar demasiados pasos en el preprocesamiento. Los apartados más debatidos y tratados fueron

**Normalizacionestandar por producto**

Por la naturaleza de las series de tiempo, cada serie posee una escala de toneladas distinta y dispar a otros productos con menos ventas. La elección fue transformar las columnas asociadas con las toneladas usando una normalización estándar (media 0 y desvío 1). Si bien se realizaron pruebas, no se eligió la normalización lineal (MinMax [0,1]) para evitar los efectos de los valores outliers.  
Algo importante a destacar es que la normalización se realizó a nivel del agrupamiento de los datos, por lo que se generó un diccionario de 780 tuplas que contenían la media y desvío para cada serie temporal con el fin de aplicar su transformación inversa en los pasos finales de la predicción. Se tuvo en cuenta además realizar dicha transformación posterior a split temporal de los datos.

[10:27](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154011)

**Tratamiento de valores faltantes o nulos**

Al inicio durante los primeros scripts el instinto natural fue reemplazar todos los valores faltantes, producto de crear lags y delta lags y reemplazarlos por 0. Lo cual se indicó como un error y que finalmente se introdujeron los datos así al modelo. Para el caso del camino de agrupamiento por <product\_id, client\_id> el tratamiento fue sustancialmente distinto ya que se tuvo en cuenta aquellos productos donde pese a no tener valores de toneladas para un cierto cliente, si lo tenía para otros, por lo que completar esos huecos con ceros era lo correcto.

[10:27](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154012)

**Indices (alta cardinalidad)**

Los índices o las variables que contienen los identificadores suelen ser excluidos de los modelos por su alta cardinalidad y no tener aparentemente un valor en el modelo final de la predicción.  
Sin embargo, esos datos suelen tener un significado intrínseco por la naturaleza del dato o por la codificación del responsable al momento de crear esos datos.  
Para el caso particular del product\_id, en un primer momento se había decidido eliminar o excluirlo del modelo, pero luego entendimos que ese código tenía una correspondencia con la cantidad de toneladas vendidas. No una relación necesariamente directa, pero lo suficientemente interesante para volver a incluirla y ejecutar un nuevo modelo.

[10:27](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154013)

**One-hot encoding para variables categóricas**

Las variables categóricas identificadas se procesaron en un pipeline considerando nuevamente la transformación para el entrenamiento y la transformación inversa para la predicción. Se realizaron algunas pruebas usando label encoding para capturar la naturaleza de las jerarquías entre los distintos productos, pero se optó finalmente por la codificación binaria.

[10:27](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154014)

**Data Leakage**

Un tema que fue abordado y no tenido en cuenta en los primeros experimentos pero que finalmente se solucionó realizando el split temporal antes de realizar cualquier transformación de los datos. Así, evitamos que datos del futuro se consideraran en el escalado de los datos de entrenamiento.

[10:28](https://austral2024.zulip.rebelare.com/" \l "narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154015)

**Clustering - Dynamic Time Warping (DTW)**

En una etapa inicial se evaluó la posibilidad de agrupar las series según la categoría o subcategoría a la que pertenecieran los productos. Sin embargo, se nos acercó una propuesta superadora como es DTW, una técnica utilizada para medir la similitud matemática y morfológica entre dos secuencias temporales que pueden variar en velocidad o duración.

Luego, hallando un número óptimo de conglomerados según la distancia o similitud entre series, podríamos entrenar las homónimas con la esperanza de encontrar las variaciones que podrían tener una serie en comparación a otros. O si alguna serie pudiese actuar de anuncio temprano de algún acontecimiento para las otras series temporales.

El primer enfoque fue aplicando el algoritmo de DTW contra el dataset agrupado por <product\_id, periodo> lo que dio un número óptimo de conglomerados de ~50 usando la técnica del codo. También en otros experimentos se usó ese agrupamiento como parte de una nueva variable. Sorprendentemente para algunos de los experimentos, figuró con una importancia relevante, por lo que se decidió crear nuevas variables para clústeres entre 5, 10, 20, 50 y 100. Finalmente, la única variable agregada fue la de la agrupación por 50 conglomerados.

En otra instancia se realizaron pruebas de cálculo de DTW contra el dataset agrupado por <product\_id, client\_id, periodo> aunque el resultado no fue el esperado al encontrar limitantes en la capacidad de cómputo dada la enorme cantidad de columnas (~700K). Es por eso, que luego se decidió acotar el problema conservando únicamente las series que tuvieran al menos tres meses consecutivos con datos distintos de cero en los meses de octubre a diciembre del 2019.

Show less

[10:28](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154016)

**Modelado con XGBoost**

El componente principal del modelo predictivo fue el módulo de regresión XGBRegressor. Dentro de los parámetros más relevantes, vale la pena considerar los que entendemos marcaron una diferencia significativa al momento de entrenar el modelo. El resto de todos los otros parámetros fueron optimizados durante una etapa de posterior:

* **max\_bin** Al controlar el número de bins o contenedores utilizados para discretizar las características continuas elegir un valor bajo o el que trae por defecto no ayuda en los modelos de regresión. En todos los experimentos usamos el máximo posible para esa combinación
* **early\_stopping\_rounds** Fue útil para prevenir el sobreajuste y encontrar el número óptimo de árboles para el modelo.

[10:28](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154017)

* **sample\_weight** Incorporado sobre el final de los experimentos, permitió que la optimización de la métrica sea en base a una ponderación, en nuestro caso penalizando aquellas predicciones para productos con mayor cantidad de toneladas que al final repercuten negativamente en la métrica del error.  
  **ESTO FUE FUNDAMENTAL**

[10:29](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154018)

* **Optimización de hiperparámetros (OPTUNA)**  
  Se eligió Optuna para automatizar y optimizar el proceso de selección de hiperparámetros ya que utiliza técnicas de optimización bayesiana y permite la búsqueda paralela y distribuida lo cual acelera ampliamente el proceso.  
  El campo de búsqueda fue alrededor de los siguientes valores y cotas:  
  'max\_depth': (3, 20)  
  'learning\_rate': (0.01, 0.3)  
  'n\_estimators': (100, 1000),  
  'min\_child\_weight': (1, 10),  
  'subsample': (0.5, 1.0),  
  'colsample\_bytree': (0.5, 1.0),  
  'reg\_alpha': (0.0, 1.0),

[10:29](https://austral2024.zulip.rebelare.com/#narrow/channel/777-l3r-Clase-04/topic/una.20muy.20buena.20solucion/near/154019)

**Evaluacion**

Utilizando múltiples semillas permitió capturar la variabilidad debida a la inicialización aleatoria en los estadios de predicción del modelo. Los experimentos tuvieron dos aproximaciones:

• **Semillerío en tiempo de predicción** Se utilizaron distinta cantidad de semillas (3, 5, 10, 50, 100) creando dicha cantidad de predicciones distintas para finalmente promediar el valor total de toneladas.

• **Semillerío en tiempo de ejecución** Se utilizaron distintas cantidades de semillas (3, 5) en los modelos con mejores resultados y cada predicción se subió a Kaggle con el fin de armar un promedio y elegir entre los modelos con el menor error (en promedio).